CLAIMS

I claim:

1) A method for developing a real-time operating system, comprising:

   a) specifying a set of *n* tasks, task(1) through task(*n*), to be scheduled for execution;

   b) specifying an algorithm for scheduling the execution of said *n* tasks; and

   c) synthesizing source code to implement a task scheduler that uses said scheduling algorithm for controlling execution of said *n* tasks.

2) The method of claim 1) further specifying *t* init-tasks that are executed only once upon initial execution of said task scheduler, *t* being less than or equal to *n*.

3) The method of claim 1) further specifying *f* f-loop tasks, each having an associated integer value l_i for *i* ranging from 1 to *f* and *f* being less than or equal to *n*, said task scheduler including a continuously executing loop such that each f-loop task executes exactly once every l_i times that the loop is executed.

4) The method of claim 1) further specifying *p* p-loop tasks, each having an associated integer value t_i for *i* ranging from 1 to *p* and *p* being less than or equal to *n*, the number t_i representing a number of regular time units, said task scheduler including a timer that schedules each p-loop task *i* to be executed approximately once every t_i time units.

5) The method of claim 1) further specifying *c* call-tasks, *c* being less than or equal to *n*, said task scheduler scheduling a call-task when another task requests that said call-task be executed.

6) The method of claim 1) further specifying *r* preemptive-tasks, *r* being less than or equal to *n*, said task scheduler including a timer mechanism that counts a specified period of time at which time if a preemptive-task is currently executing, the task's state is stored and execution is given to said task scheduler to schedule another task

until a later time when the task scheduler restores the state of said preemptive-task and execution of said preemptive-task is continued.

7) The method of claim 1) where tasks are given priority values such that whenever the task scheduler chooses between scheduling multiple tasks, all of which being ready to be executed, said task scheduler chooses from among those tasks that have the highest priority values.

8) An apparatus for developing a real-time operating system comprising

A computer;

A software synthesis program on said computer, wherein said software synthesis program comprises:

a) means for specifying a set of *n* tasks, task(1) through task(*n*), to be scheduled for execution;

b) means for specifying an algorithm for scheduling the execution of said *n* tasks; and

c) means for synthesizing source code to implement a task scheduler that uses said scheduling algorithm for controlling execution of said *n* tasks.

9) The apparatus of claim 8) including means for specifying *t* init-tasks that are executed only once upon initial execution of said task scheduler, *t* being less than or equal to *n*.

10) The apparatus of claim 8) including means for specifying *f* f-loop tasks, each having have an associated integer value l_i for *i* ranging from 1 to *f* and *f* being less than or equal to *n*, said task scheduler including a continuously executing loop such that each f-loop task executes exactly once every l_i times that the loop is executed.

11) The apparatus of claim 8) including means for specifying *p* p-loop tasks, each having an associated integer value t_i for *i* ranging from 1 to *p* and *p* being less than or equal to *n*, the number t_i representing a number of regular time units, said task scheduler including a timer that schedules each p-loop task *i* to be executed approximately once every t_i time units.

12) The apparatus of claim 8) including means for specifying *c* call-tasks, *c* being less than or equal to *n*, said task scheduler scheduling a call-task when another task requests that said call-task be executed.

13) The apparatus of claim 8) including means for specifying *r* preemptive-tasks, *r* being less than or equal to *n*, said task scheduler including a timer mechanism that counts a specified period of time at which time if a preemptive-task is currently executing, the task's state is stored and execution is given to said task scheduler to schedule another task until a later time when the task scheduler restores the state of said preemptive-task and execution of said preemptive-task is continued.

14) The apparatus of claim 8) where tasks are given priority values such that whenever the task scheduler chooses between scheduling multiple tasks, all of which are ready to be executed, said task scheduler chooses from among those tasks that have the highest priority values.